



Home



Search



List

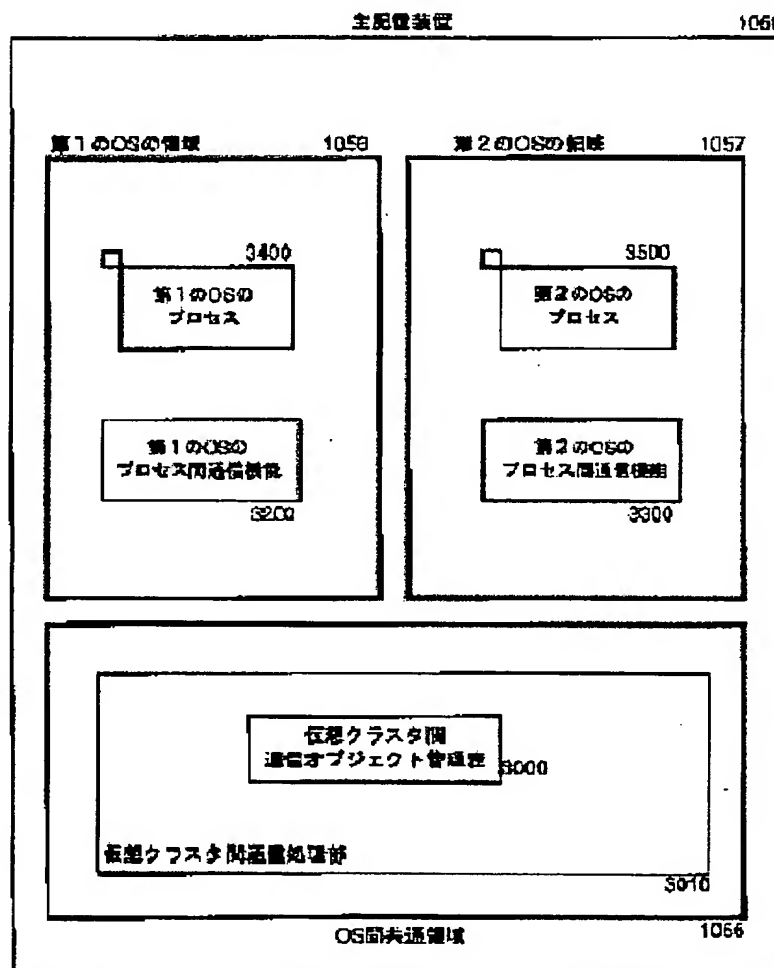
☐ Include

MicroPatent® PatSearch FullText: Record 1 of 1

Search scope: JP (bibliographic data only)

Years: 1991-2003

Patent/Publication No.: JP11085547

[Order This Patent](#)[Family Lookup](#)[Find Similar](#)[Legal Status](#)[Go to first matching text](#)

JP11085547 A
 VIRTUAL CLUSTER CONFIGURATION METHOD
 HITACHI LTD

Inventor(s): INOUE TARO ; ARAI TOSHIKI

Application No. 09248179 JP09248179 JP, Filed 19970912, A1 Published 19990330

Abstract: PROBLEM TO BE SOLVED: To reduce exclusive control overhead in a high multiple symmetrical multiprocessor system(SMP) by providing a process that operates on a different virtual cluster with an inter- virtual cluster communicating means which mutually communicates.

SOLUTION: Two virtual clusters are configured on an SMP that has two processors, and a main storage device 1050 consists of an inter-OS common area 1055 and areas 1056 and 1057 for 1st and 2nd OSs. An inter- virtual cluster processing part 3010 exists in the area 1050, and an inter-virtual cluster communication object management table

3000 exists in it. Inter-process communication functions 3200 and 3300 for the 1st and 2nd OSs exist in the areas 1056 and 1057 of the 1st and 2nd OSs respectively. They perform inter-process communication processing to a process on which respective OSs operate, themselves. Respective processes 3400 and 3500 for the 1st and 2nd OSs exist in the areas 1056 and 1057 of the 1st and 2nd OSs.

Int'l Class: G06F00946; G06F01516

Patents Citing this One: No US, EP, or WO patents/search reports have cited this patent.

[Home](#)[Search](#)[List](#)

For further information, please contact:
[Technical Support](#) | [Billing](#) | [Sales](#) | [General Information](#)

【特許請求の範囲】

【請求項 1】プロセッサ、物理メモリ、外部デバイス等のハードウェア資源を専有しその上でオペレーティングシステムが動作する複数の仮想クラスタを 1 台の計算機上に構成する仮想クラスタ構成手段を有し、異なる仮想クラスタ上で動作するプロセスが相互に通信する仮想クラスタ間通信手段を有することを特徴とする仮想クラスタ構成方法。

【請求項 2】請求項 1 の仮想クラスタ構成方法において、当該仮想クラスタ構成手段は、第 1 の仮想クラスタのオペレーティングシステム（第 1 のオペレーティングシステム）の起動時に、他の仮想クラスタが利用するハードウェア資源（プロセッサ、物理メモリ、外部デバイス）を他の仮想クラスタのために予約して第 1 のオペレーティングシステムからアクセス不可能にする手順を有し、予約された物理メモリ領域に第 2 の仮想クラスタのオペレーティングシステム（第 2 のオペレーティングシステム）をロードして起動する手順を有することを特徴とする仮想クラスタ構成方法。

【請求項 3】請求項 2 の仮想クラスタ構成方法において、仮想クラスタが専有するデバイスからの割り込みを、当該仮想クラスタが専有するプロセッサに通知する手段を有することを特徴とする仮想クラスタ構成方法。

【請求項 4】請求項 1 の仮想クラスタ構成方法において、前記の仮想クラスタ間通信手段は、仮想クラスタ間通信オブジェクトの名前と、第 1 のオペレーティングシステムのプロセス間通信機能におけるプロセス間通信オブジェクトハンドルと、第 2 のオペレーティングシステムのプロセス間通信機能におけるプロセス間通信オブジェクトハンドルとの対応を管理する対応管理手段を有することを特徴とする仮想クラスタ構成方法。

【請求項 5】請求項 4 の仮想クラスタ構成方法において、第 1 のオペレーティングシステム上のプロセスが、第 2 のオペレーティングシステム上のプロセスとの通信要求を発行した場合、当該通信要求を前記対応管理手段を用いて第 1 のオペレーティングシステムのプロセス間通信機能への通信要求、あるいは第 2 のオペレーティングシステムのプロセス間通信機能への通信要求に変換する手順を有することを特徴とする仮想クラスタ構成方法。

【請求項 6】請求項 1 の仮想クラスタ構成方法において、ある仮想クラスタ上で動作するプロセスを、他の仮想クラスタへ移動させて動作させるプロセスマイグレーション手段を有することを特徴とする仮想クラスタ構成方法。

【請求項 7】請求項 6 の仮想クラスタ構成方法において、

前記のプロセスマイグレーション手段は、各仮想クラスタのプロセッサ利用率の情報を得る手順と、得られたプロセッサ利用率の情報から、最もプロセッサ利用率が低い仮想クラスタをプロセスのマイグレーション先に選択する手順とを有することを特徴とする仮想クラスタ構成方法。

【請求項 8】請求項 6 の仮想クラスタ構成方法において、

前記のプロセスマイグレーション手段は、マイグレーション対象プロセスの親プロセスが動作する仮想クラスタをプロセスのマイグレーション先に選択する手順を有することを特徴とする仮想クラスタ構成方法。

【請求項 9】請求項 6 の仮想クラスタ構成方法において、

前記のプロセスマイグレーション手段は、マイグレーション対象プロセスの親プロセスが動作するのは異なる仮想クラスタをプロセスのマイグレーション先に選択する手順を有することを特徴とする仮想クラスタ構成方法。

【請求項 10】請求項 6 の仮想クラスタ構成方法において、

前記のプロセスマイグレーション手段は、各仮想クラスタの実メモリ利用率の情報を得る手順と、得られた実メモリ利用率の情報から、最も実メモリ利用率が低い仮想クラスタをプロセスのマイグレーション先に選択する手順とを有することを特徴とする仮想クラスタ構成方法。

【請求項 11】請求項 6 の仮想クラスタ構成方法において、

前記のプロセスマイグレーション手段は、マイグレーション先の仮想クラスタをランダムに選択する手順を有することを特徴とする仮想クラスタ構成方法。

【請求項 12】請求項 6 の仮想クラスタ構成方法において、

前記のプロセスマイグレーション手段は、マイグレーション先の仮想クラスタをラウンドロビンに選択する手順を有することを特徴とする仮想クラスタ構成方法。

【請求項 13】請求項 1 の仮想クラスタ構成方法において、

仮想クラスタ間でデバイスを共有するデバイス共有手段を有することを特徴とする仮想クラスタ構成方法。

【請求項 14】請求項 1 の仮想クラスタ構成方法において、

仮想クラスタ間でメモリを共有する手段を有することを特徴とする仮想クラスタ構成方法。

【請求項 15】請求項 1 の仮想クラスタ構成方法において、

ある 1 つの仮想クラスタを別の仮想クラスタのホットス

タンバイとして動作させる手段を有することを特徴とする仮想クラスタ構成方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、計算機の制御方法に関わる。特に、1台計算機上で、複数のオペレーティングシステムを動作させ、各々のオペレーティングシステム上のプロセス同士が通信する方法に関する。

【0002】

【従来の技術】最近の計算機システムでは、処理能力の向上を図るために複数のプロセッサ(CPU)を搭載するマルチプロセッサ構成が一般的となっている。そのマルチプロセッサシステムの構成方式として、対称型マルチプロセッサシステム(SMP)とクラスタ型システムがある。

【0003】SMPは、システムを構成するすべてのCPUがメモリを共有し、すべての論理的な資源(プロセス、スレッド等)および物理的な資源(CPU、物理メモリ、外部デバイス等)を共有する。そして、1つのオペレーティングシステム(OS)がシステム全体を制御する。

【0004】一方、クラスタ型システムでは、システム全体が複数のクラスタに分割される。各クラスタはCPU、メモリ、デバイスで構成され、各クラスタ上のOSが当該クラスタを制御する。この場合は、論理的な資源、物理的な資源の双方とも共用はされず、完全に分離される。

【0005】また、従来から、1台の計算機上に複数の仮想的な計算機(仮想計算機:VM)を実現する仮想計算機システムがある。仮想計算機システムでは、仮想計算機制御プログラムが計算機のすべての資源を管理し仮想化することにより仮想計算機を構成する。仮想計算機システムを実現する場合、すべての計算機資源の仮想化を行うため、ハードウェアおよびソフトウェア(仮想計算機制御プログラム)の規模が大きくなるという問題がある。

【0006】

【発明が解決しようとする課題】先に述べたように、対称型マルチプロセッサシステム(SMP)では、すべてのシステムの資源を共用する。例えば、メモリはすべてのCPUから共有されるので、複数のCPUからアクセスされる可能性があるメモリ領域については排他制御が必要となる。このオーバーヘッドはCPU数のベキ乗に比例して増加するため、高多重SMPの実現の上での大きな問題となる。

【0007】そこで、SMPのハードウェア資源を分割専有する仮想クラスタシステムであれば、SMPにおけるCPU数が増加しても排他制御オーバーヘッドの増加を小さくできる。例えば、8台のCPUを有するSMPをそのまま使用する場合と、4台のCPUを専有する2つ

の仮想クラスタに分割する場合の排他制御オーバーヘッドを比べると、先に述べたように、排他制御オーバーヘッドはCPU数のベキ乗に比例するので、2つの仮想クラスタに分割した場合の方が排他制御オーバーヘッドは小さくなる。

【0008】本発明の第1の目的は、高多重の対称型マルチプロセッサシステム(SMP)において、排他制御オーバーヘッドを小さくするために、仮想クラスタを構成する方法を提供することにある。

【0009】SMPの別の問題点は、バスの利用率である。バスには各種のデータが流れる。キャッシュミスが発生した場合のメモリからキャッシュへのデータ、各CPUのキャッシュの内容の不一致を起こさないようにするためのコヒーレントトランザクション等である。SMPの性能が飽和する原因としては、バスの利用率が過大になるバスネックの発生がある。バスネックの発生を防止するためには、先に挙げたようなバスの利用を減らしバスの利用率を低くする必要がある。そこで、仮想クラスタを構成すれば、プロセスが動作するプロセッサが限定されるので、キャッシュミスの発生を減少できる。また、メモリは分割専有されるために他の仮想クラスタからのメモリアccessはないので、コヒーレントトランザクションの波及する範囲を当該仮想クラスタ中に限定できる。

【0010】本発明の第2の目的は、高多重の対称型マルチプロセッサシステムにおいて、キャッシュミス、およびコヒーレントトランザクションを減らすために、仮想クラスタを構成する方法を提供することにある。

【0011】上記のように仮想クラスタを構成した場合、仮想クラスタ同士が連携して処理をすすめようとすると、仮想クラスタ間でのプロセスの通信が必要となる。

【0012】本発明の第3の目的は、仮想クラスタシステムにおいて、各仮想クラスタ上のプロセスが相互に通信する方法を提供することにある。

【0013】

【課題を解決するための手段】本発明では、第1の仮想クラスタのオペレーティングシステム(第1のオペレーティングシステム)の起動時に、他の仮想クラスタが利用するハードウェア資源を他の仮想クラスタのために予約して第1のオペレーティングシステムの管理対象から外して第1のオペレーティングシステムからはアクセス不可能にすることにより、1台の計算機において複数の仮想クラスタを、特別なハードウェアなしに構成することが可能となる。

【0014】また、本発明では、仮想クラスタ間通信オブジェクトの名前と、第1のオペレーティングシステムのプロセス間通信機能におけるプロセス間通信オブジェクトハンドルと、第2のオペレーティングシステムのプロセス間通信機能におけるプロセス間通信オブジェクト

ハンドルとの対応管理手段を設ける。この対応管理手段により、第1のオペレーティングシステム上のプロセスおよび第2のオペレーティングシステム上のプロセスから発行された仮想クラスタ間通信オブジェクトの名前を指定した通信要求を、第1のオペレーティングシステムあるいは第2のオペレーティングシステムのプロセス間通信機能への要求へと変換できる。これにより、例えば、イベント同期によるプロセス間通信の場合では、ウェイトしている第1のオペレーティングシステム上のプロセスを第2のオペレーティングシステム上のプロセスがポストし、ウェイトしている第2のオペレーティングシステム上のプロセスを第1のオペレーティングシステム上のプロセスがポストすることが可能となる。

【0015】

【発明の実施の形態】以下、本発明の実施例を、図面により詳細に説明する。

【0016】以下に説明する実施例では、2台のプロセッサを持つ対称型マルチプロセッサシステム(SMP)上で、2つの仮想クラスタを構成し、各々の仮想クラスタは1個ずつのプロセッサを専有し、その上では第1のOSおよび第2のOSが稼働するものとする。もちろん、設定する仮想クラスタ数はいくつでも構わない。ただし、対称型マルチプロセッサシステムが持つプロセッサの個数より多くの仮想クラスタを構築することはできない。

【0017】図1は、本発明の実施例を実施する計算機システムである対称型マルチプロセッサシステム1000の構成を示す図である。対称型マルチプロセッサシステム1000は、プロセッサ1010、プロセッサ1011、主記憶装置1050、クロック1060、ブートROM1070、割り込み制御装置1080、バス1090、割り込み信号線1100、および割り込みバス1101から成り、キーボード装置1110、モニタ装置1120、磁気ディスク装置1130、その他の外部装置1140が接続される。

【0018】割り込み信号線1100は、割り込み制御装置1080と各種の割り込みを発生する装置を接続する。割り込みが発生すると、割り込み信号線1100を経由して割り込み制御装置1080に通知され、割り込み制御装置1080はそれを数値化して、割り込みバス1101によりプロセッサ1010およびプロセッサ1011に通知する。

【0019】割り込み制御装置1080は、外部の装置やクロック1060からの割り込み要求信号を数値化してプロセッサ1010およびプロセッサ1011に伝える。プロセッサからの指示により特定の装置からの割り込みをプロセッサに通知しないようにすることもできる。

【0020】クロック1060は、周期的にクロック割り込みを発生する。

【0021】ブートROM1070には、この対称型マルチプロセッサシステム1000が起動された時の起動手順の最初の一部が記録されている。

【0022】プロセッサ1010およびプロセッサ1011は、各々、演算装置1020、割り込みテーブルレジスタ1030、ページテーブルレジスタ1040、およびアドレス変換装置1045から成る。以下では、プロセッサ1010はCPU0、プロセッサ1011はCPU1として示す場合がある。この図では、プロセッサはプロセッサ1010およびプロセッサ1011の2台のみが書かれているが、何台でもかまわない。割り込みテーブルレジスタ1030は、当該プロセッサを専有する仮想クラスタのOSの割り込みテーブル1051のアドレスを格納している。ページテーブルレジスタ1040は、当該プロセッサを専有する仮想クラスタのOSのページテーブル1052のアドレスを格納している。アドレス変換装置1045は、演算装置1020からアドレスを受け取り、ページテーブルレジスタ1040が示すページテーブル1052に基づいて仮想アドレスから実アドレスへの変換を行う。演算装置1020は変換された実アドレスを用いて主記憶装置1050へアクセスする。

【0023】対称型マルチプロセッサシステム1000には、外部装置として、キーボード装置1110、モニタ装置1120、磁気ディスク装置1130、その他の外部装置1140が接続されており、モニタ装置1120を除いては、割り込みバス1100に接続される。

【0024】主記憶装置1050には、第1のオペレーティングシステム(OS)の領域1056、第2のOSの領域1057、およびOS間共通領域1055がある。第1のOSの領域1056および第2のOSの領域1057には、それぞれのOSのページテーブル1052および割り込みテーブル1051がある。OS間共通領域1055には、仮想クラスタ間通信処理部3010がある。

【0025】図2は、ページテーブル1052の構成を示す図である。ページテーブル1052は仮想アドレス空間単位に存在し、仮想ページ番号1200毎にエントリがあり、そのエントリは有効ビット1210および物理ページ番号1220で構成される。有効ビット1210は当該仮想ページに対して物理ページが割り当てられているか否かを示す。図2の場合、仮想ページ1は有効ビット1210がセットされているので、物理ページが割り当てられているが、仮想ページ4は有効ビット1210がセットされていないので、物理ページが割り当てられていない。物理ページ番号1220は当該仮想ページに割り当てられている物理ページの番号を示す。

【0026】アドレス変換装置1045は、演算装置1020から仮想アドレスを受け取り、ページテーブルレジスタ1040が示すページテーブル1052の内容に

基づいて仮想アドレスから実アドレスへの変換を行い、得られた実アドレスを演算装置1020へ返す。演算装置1020はこの実アドレスを用いて主記憶装置1050へアクセスする。

【0027】ページテーブルレジスタ1040によりページテーブルを切り替えることで、独立したアドレス空間を構築できる。即ち、第1のOSの空間および第2のOSの空間を構築できる。そして、各々のOSのページテーブルにおいて、共通領域に対応する仮想ページに対して同じ物理ページを設定すればOS間共通領域を実現できる。

【0028】図3は、割り込みテーブル1051の構成を示す図である。割り込みテーブル1051は、割り込み番号とそれに対する割り込み処理ハンドラのアドレスの対応を格納する。割り込みテーブル1051は、割り込み番号1300毎にエントリがあり、そのエントリは割り込みハンドラアドレス1310で構成される。制御装置1080から割り込みを受けたプロセッサは、割り込みテーブルレジスタ1030の内容が示す割り込みテーブル1051の中から割り込み番号1300に対応する割り込みハンドラアドレス1310を得て、そこに制御を渡して割り込み処理を行う。

【0029】図4は、計算機のブート時に実行されるカーネルローダ1131の処理のフローである。カーネルローダ1131は、システムの起動時にブートROM1070に格納される起動プログラムにより磁気ディスク装置1130から主記憶にロードされ実行を開始する。まず、主記憶リスト1710、デバイスリスト1720、ロードオブジェクトリスト1740を初期化する(ステップ1410)。これらのリストの詳細は後で説明する。カーネルが使用するページテーブル領域を割り当てる(ステップ1420)。この計算機に接続される計算機のハードウェア構成を検査し(ステップ1430)、それに基づいてデバイスリスト1720を作成する(ステップ1440)。次に、磁気ディスク装置1130に格納されるカーネル構成情報ファイル1600を読み込んでOSのカーネル構成情報1900とする(ステップ1450)。カーネル構成情報1900の詳細は後で説明する。カーネル構成情報1900で指定されたファイルを読み込み、ロードオブジェクトリスト1740を追加する(ステップ1460)。そして、ページテーブルレジスタ1030に構築したページテーブルのアドレスをセットして仮想アドレスモードへ移行し(ステップ1470)、構築した主記憶リスト1710、デバイスリスト1720、ロードオブジェクトリスト1740、カーネル構成情報アドレス1730から成るパラメータテーブル1700をパラメータとしてOSの初期化処理へ制御を渡す(ステップ1480)。

【0030】図5は、カーネル構成情報ファイル1600の内容を示す図である。カーネル構成情報ファイル1

600の内容がそのままカーネル構成情報1900となる。カーネル構成情報ファイル1600には、オペレーティングシステムが使用するデータを格納している。データにはその種別を示す名前が付けられており、オブジェクトファイル1610という名前のエントリにはそのデータ1630が格納され、第2のOS1620という名前のエントリには第2のOSが使用するデータ1640が格納される。第2のOSが使用するデータ1640には、第2のOS(即ち、第2の仮想クラスタ)に割り当てるプロセッサ数、主記憶の大きさ、デバイスの情報等が含まれる。

【0031】このカーネル構成情報ファイル1600を書き換えることにより、仮想クラスタの構成を変更することが可能である。

【0032】図6は、主記憶リスト1710、デバイスリスト1720、ロードオブジェクトリスト1740およびパラメータテーブル1700の構成を示す図である。パラメータテーブル1700は、主記憶リスト1710、デバイスリスト1720、ロードオブジェクトリスト1740、およびカーネル構成情報アドレス1730で構成される。

【0033】主記憶リスト1710は、主記憶ブロック1750のリストである。主記憶ブロック1750は、連続した主記憶領域の状況を記述する。ベースアドレス1760は当該主記憶ブロックの開始物理アドレスであり、サイズ1770はその大きさを示す。利用状況1780は当該主記憶ブロックが未使用か否かを示す値が格納され、次エントリ1785には次の主記憶ブロックへのポインタが格納される。

【0034】デバイスリスト1720は、デバイスブロック1800のリストである。デバイスブロック1800は、カーネルローダが検出したハードウェアの情報が格納される。デバイスタイプ1810には当該デバイスの種類が格納され、デバイス情報1820には当該デバイスに関する情報、例えば、割り込み番号やI/Oアドレス等は格納される。次エントリ1830には次のデバイスブロックへのポインタが格納される。

【0035】ロードオブジェクトリスト1740は、ロードオブジェクトブロック1850のリストである。ロードオブジェクトブロック1850は、カーネルが主記憶にロードしたオブジェクトファイルのデータを保持する。オブジェクトファイル名1860は当該ロードオブジェクトブロックで記述されるオブジェクトファイルのファイル名が格納され、オブジェクトアドレス1780は当該ロードオブジェクトブロックで記述されるオブジェクトファイルが格納されるカーネル空間のアドレスが格納される。

【0036】カーネル構成情報アドレス1730はカーネル構成情報1900のアドレスが格納される。

【0037】図7は、割り込みベクタ管理テーブル20

00の構造を示す図である。割り込みベクタ管理テーブル2000は、OS毎にあり、各割り込み番号に関してその割り込み番号を当該OSが使用するか否かを示す。カーネルは、デバイスドライバの初期化時に割り込み番号の使用を要求した場合にこの割り込みベクタ管理テーブル2000を調べ、使用中でない場合には要求された割り込み番号を使用中にして、その割り込み番号を使用する権利を与える。

【0038】図8は、I/Oアドレス管理リスト2100の構造を示す図である。I/Oアドレス管理リスト2100は、OS毎にあり、I/Oアドレスの範囲を示すエントリ2110と次エントリへのポインタ2120から成り、デバイスドライバの初期化時にI/Oアドレスの使用を要求した場合にこのI/Oアドレス管理リスト2100を調べ、使用中でない場合には当該I/Oアドレスのエントリを追加して、そのI/Oアドレスを使用する権利を与える。

【0039】図9は、プロセッサ管理テーブル2150の構造を示す図である。プロセッサ管理テーブル2150は、OS毎にあり、各プロセッサに関して当該OSがそのプロセッサを使用するか否かを示す。

【0040】図10は、第1のOSの初期化処理のフローである。まず、パラメータとして渡されたパラメータテーブル1700中のロードオブジェクトリスト1740を参照して、カーネルローダ1131がロードしたオブジェクトファイルの外部参照アドレスを解決する(ステップ2200)。次に、カーネル構成情報1900の中の第2のOS1620という名前のエントリの第2のOSが使用するデータ1640から、第2のOSのために確保する主記憶の大きさを求め、その分を除いて主記憶リスト1710を構成しなおして第2のOSの主記憶を確保する(ステップ2210)。これにより、第1のOSからは第2のOSのために確保した領域は見えなくなる。カーネル内部のデータ構造を初期化する(ステップ2220)。次に、第1のOSが管理する割り込みベクタ管理テーブル2000およびI/Oアドレス管理リスト2100に対して、第2のOSが使用するデバイス(カーネル構成情報1900の中の第2のOS1620という名前のエントリの第2のOSが使用するデータ1640に格納されている)を登録して第2のOSのデバイスを予約する(ステップ2230)。これにより予約した第2のOSのデバイスは第1のOSからは利用できなくなる。第1のOSが管理するプロセッサ管理テーブル2150に対して、第2のOSが使用するプロセッサ(カーネル構成情報1900の中の第2のOS1620という名前のエントリの第2のOSが使用するデータ1640に格納されている)を登録して第2のOSのプロセッサを予約する(ステップ2235)。これにより予約した第2のOSのプロセッサは第1のOSからは利用できなくなる。カーネルが直接管理するシステムデバ

スを初期化する(ステップ2240)。ここでは、第1のOSが属する仮想クラスタが専有するプロセッサのうち、この初期化処理を実行中のプロセッサ以外のプロセッサに対して初期化割り込みをかける。更に、割り込み制御装置1080の初期化も行う。即ち、カーネル構成情報1900の中の第2のOS1620という名前のエントリの第2のOSが使用するデータ1640を参照して、第2のOSが管理するデバイスの割り込みが、第2のOSが専有するプロセッサに送られるように設定する。カーネルローダ1131がロードしたオブジェクトファイルを初期化し(ステップ2250)、初期プロセスを作成する(ステップ2260)。

【0041】図11は、第2のOSのロード処理のフローである。第2のOSの物理メモリ領域は第1のOSからはアクセスできないので、第2のOSに割り当てた物理メモリ領域を第1のOSの仮想空間に一時的にマッピングする(ステップ2300)。マッピングした領域に第1にOSのファイル読み込み手順を利用して第2のOSのオブジェクトファイルを読み込む(ステップ2310)。次に、第2にOSのためのページテーブルを作成する(ステップ2320)。この時、第1のOS共有する部分については第2のOSからも参照できるようにページテーブルを構築する。第2のOSの外部参照を解決する(ステップ2330)。そして、ステップ2300で設定したマッピングを解除する(ステップ2340)。ページテーブルアドレスとスタックポインタから成るOSコンテキストを設定し(ステップ2350)、第2のOSが専有するプロセッサに初期化割り込みを送信し(ステップ2355)、第2のOSの初期化手順を実行する(ステップ2360)。最後に、現在の割り込みテーブルを書き換える(ステップ2370)。

【0042】図12は、割り込み制御装置1080を説明する図である。割り込み制御装置1080は、各デバイスからの割り込みをどのプロセッサあるいはプロセッサ群へ通知するかを設定する機能を有する。この割り込み制御装置1080により、デバイスからの割り込みを、当該デバイスが属する仮想クラスタが専有するプロセッサへ通知することが可能となる。

【0043】割り込みマスクレジスタ2440はデバイスで発生した割り込みをプロセッサに通知してよいかどうかを設定される。これはプロセッサ1010からの命令で設定できる。選択装置2430は、割り込み信号線1100から割り込み要求を受けた時と割り込みマスクレジスタ2440の内容が変更された時に、選択装置2430が記憶している未処理割り込みと割り込みマスクレジスタ2440の内容を比較して、割り込みマスクレジスタ2440に割り込み可能と設定されていて最も優先度の高い割り込みから順にプロセッサに通知する。

【0044】割り込み配送テーブル2410は、それぞれのデバイスについてどのプロセッサあるいはプロセッ

サ群へ通知するかを示す配送先2411と、通知するときの割り込み番号2412から成る。この割り込み配送テーブル2410の内容はプロセッサの命令により自由に設定できる。

【0045】割り込み送信装置2420は、選択装置2430からの信号を受けると、割り込み配送テーブル2410の内容にしたがって割り込み通知先と割り込み番号を決定し、その信号を割り込みバス1101に送信する。

【0046】以上により、1台の対称型マルチプロセッサシステム1000の上に、プロセッサ、主記憶およびデバイスを専有しその上でOSが動作する仮想クラスタを複数個構築することが可能となる。

【0047】以下では、異なる仮想クラスタ上で動作するプロセスが相互に通信する仮想クラスタ間通信について説明する。

【0048】図13は、主記憶装置1050における仮想クラスタ間通信関連の内容を示す図である。主記憶装置1050は、図1に示したように、OS間共通領域1055、第1のOSの領域1056、および第2のOSの領域1057から成る。OS間共通領域1055の中には、仮想クラスタ間通信処理部3010があり、その中には仮想クラスタ間通信オブジェクト管理表3000がある。第1のOSの領域1056および第2のOSの領域1057には、それぞれ第1のOSのプロセス間通信機能3200および第2のOSのプロセス間通信機能3300がある。これらは、それぞれのOS自身がその上で動作するプロセスに対するプロセス間通信処理を行う部分である。そして、第1のOSの領域1056および第2のOSの領域1057には、それぞれ第1のOSのプロセス3400および第2のOSのプロセス3500がある。

【0049】図14は、仮想クラスタ間通信オブジェクト管理表3000の図である。仮想クラスタ間通信オブジェクト管理表3000は、エントリ有効フラグ3010、通信オブジェクトの名前3020、第1のOSの通信オブジェクトハンドル3030、第1のOSの通信オブジェクトハンドル有効フラグ3040、第2のOSの通信オブジェクトハンドル3050、第2のOSの通信オブジェクトハンドル有効フラグ3060から成る。

【0050】通信オブジェクトの名前3020には、仮想クラスタ間通信オブジェクトの名前が格納される。エントリ有効フラグ3010は、このエントリが有効か無効かを示すフラグである。

【0051】第1のOSの通信オブジェクトハンドル3030には、第1のOSのプロセス間通信機能3200のオブジェクトハンドルを格納し、第1のOSの通信オブジェクトハンドル有効フラグ3040はそれの有効か無効かを示すフラグである。

【0052】第2のOSの通信オブジェクトハンドル3

050には、第2のOSのプロセス間通信機能3300のオブジェクトハンドルを格納し、第2のOSの通信オブジェクトハンドル有効フラグ3060はそれの有効か無効かを示すフラグである。

【0053】以下では、イベント同期通信（ポスト/ウェイト）の場合を例にとって説明を進めるが、これ以外の通信（メッセージ、セマフォ等）でも同様である。

【0054】図15は、第1のOSのプロセス3400が発行したイベント同期通信のオープン要求システムコールの処理のフローである。このシステムコールは、イベントの名前をパラメータに指定してオープン要求を行い、その結果として記述子を得る。以降のイベント同期通信関連のシステムコールでは、この記述子を用いることになる。

【0055】まず、仮想クラスタ間通信オブジェクト管理表3000を調べて、エントリ有効フラグ3010がONで、かつ、指定されたイベントの名前と通信オブジェクトの名前3020が同じであるエントリを探す（ステップ4010）。仮想クラスタ間通信オブジェクト管理表3000にこのようなエントリがあれば、既にこの名前のイベントはオープンされていることになる。そこで、このエントリについて、第1のOSの通信オブジェクトハンドル有効フラグ3040を調べ（ステップ4090）、ONなら、そのエントリの記述子を返して終了する（ステップ4100）。OFFなら、第1のOSのプロセス間通信機能3200のオープンのシステムコールをこのイベントの名前で発行することにより、第1のOSのプロセス間通信機能3200におけるオブジェクトハンドルを獲得し（ステップ4110）、獲得したオブジェクトハンドルを第1のOSの通信オブジェクトハンドル3030にセットし（ステップ4120）、第1のOSの通信オブジェクトハンドル有効フラグ3040をONにし（ステップ4130）、そのエントリの記述子を返して終了する（ステップ4140）。

【0056】一方、ステップ4010で、エントリ有効フラグ3010がONで、かつ、指定されたイベントの名前と通信オブジェクトの名前3020が同じであるエントリが、仮想クラスタ間通信オブジェクト管理表3000になければ、エントリ有効フラグ3010がOFFのエントリを探す（ステップ4020）。なければ、仮想クラスタ間通信オブジェクト管理表3000のすべてのエントリが使用中であることになるのでエラーリターンする（ステップ4030）。エントリ有効フラグ3010がOFFのエントリがあれば、第1のOSのプロセス間通信機能3200のオープンのシステムコールをこのイベントの名前で発行することにより、第1のOSのプロセス間通信機能3200のオブジェクトハンドルを獲得し（ステップ4040）、獲得したオブジェクトハンドルを第1のOSの通信オブジェクトハンドル3030にセットし（ステップ4050）、第1のOSの通信

オブジェクトハンドル有効フラグ3040をONにし(ステップ4060)、パラメータで指定されたイベントの名前を通信オブジェクトの名前3020に格納し(ステップ4065)、エントリ有効フラグ3010をONにして(ステップ4070)、そのエントリの記述子を返して終了する(ステップ4080)。

【0057】図16は、第1のOSのプロセス3400が発行したイベント同期通信のウェイト要求システムコールの処理のフローである。このシステムコールは、記述子をパラメータに指定してウェイト要求を行う。まず、当該ウェイト要求において指定された記述子に対応するエントリを仮想クラスタ間通信オブジェクト管理表3000からみつけ、そのエントリにおける第1のOSの通信オブジェクトハンドル3030を用いて、第1のOSのプロセス間通信機能3200のウェイト要求のシステムコールを発行し(ステップ4210)、リターンする(ステップ4220)。

【0058】図17は、第1のOSのプロセス3400が発行したイベント同期通信のポスト要求システムコールの処理のフローである。このシステムコールは、記述子をパラメータに指定してポスト要求を行う。まず、第2のOSへコンテキスト切替えを行い(ステップ4310)、当該ポスト要求において指定されたイベント記述子に対応するエントリを仮想クラスタ間通信オブジェクト管理表3000からみつけ、そのエントリにおける第2のOSの通信オブジェクトハンドル3050を用いて、第2のOSのプロセス間通信機能3300のポスト要求のシステムコールを発行する(ステップ4320)。そして、第1のOSへコンテキスト切替えを行い(ステップ4330)、リターンする(ステップ4340)。

【0059】図18は、第1のOSのプロセス3400が発行したイベント同期通信のクローズ要求システムコールの処理のフローである。このシステムコールは、記述子をパラメータに指定してクローズ要求を行う。まず、当該クローズ要求において指定された記述子に対応するエントリを仮想クラスタ間通信オブジェクト管理表3000からみつけ、そのエントリにおける第1のOSの通信オブジェクトハンドル3030を用いて、第1のOSのプロセス間通信機能3200のクローズ要求のシステムコールを発行し(ステップ4410)、第1のOSの通信オブジェクトハンドル有効フラグ3040をOFFにする(ステップ4420)。第2のOSの通信オブジェクトハンドル有効フラグ3060がOFFなら、エントリ有効フラグ3010をOFFにして当該エントリを無効にして(ステップ4440)、リターンする(ステップ4450)。一方、第2のOSの通信オブジェクトハンドル有効フラグ3060がONなら、リターンする(ステップ4450)。

【0060】図19は、第2のOSのプロセス3500

が発行したイベント同期通信のオープン要求システムコールの処理のフローである。このシステムコールは、イベントの名前をパラメータに指定してオープン要求を行い、その結果として記述子を得る。以降のイベント同期通信関連のシステムコールでは、この記述子を用いることになる。

【0061】まず、仮想クラスタ間通信オブジェクト管理表3000を調べて、エントリ有効フラグ3010がONで、かつ、指定されたイベントの名前と通信オブジェクトの名前3020が同じであるエントリを探す(ステップ4510)。仮想クラスタ間通信オブジェクト管理表3000にこのようなエントリがあれば、既にこの名前のイベントはオープンされていることになる。そこで、このエントリについて、第2のOSの通信オブジェクトハンドル有効フラグ3060を調べ(ステップ4590)、ONなら、そのエントリの記述子を返して終了する(ステップ4600)。OFFなら、第2のOSのプロセス間通信機能3300のオープンのシステムコールをこのイベントの名前で発行することにより、第2のOSのプロセス間通信機能3300のオブジェクトハンドルを獲得し(ステップ4610)、獲得したオブジェクトハンドルを第2のOSの通信オブジェクトハンドル3050にセットし(ステップ4620)、第2のOSの通信オブジェクトハンドル有効フラグ3060をONにし(ステップ4630)、そのエントリの記述子を返して終了する(ステップ4640)。

【0062】一方、ステップ4010で、エントリ有効フラグ3010がONで、かつ、指定されたイベントの名前と通信オブジェクトの名前3020が同じであるエントリが、仮想クラスタ間通信オブジェクト管理表3000になければ、エントリ有効フラグ3010がOFFのエントリを探す(ステップ4520)。なければ、仮想クラスタ間通信オブジェクト管理表3000のすべてのエントリが使用中であることになるのでエラーリターンする(ステップ4530)。エントリ有効フラグ3010がOFFのエントリがあれば、第2のOSのプロセス間通信機能3300のオープンのシステムコールをこのイベントの名前で発行することにより、第2のOSのプロセス間通信機能3300のオブジェクトハンドルを獲得し(ステップ4540)、獲得したオブジェクトハンドルを第2のOSの通信オブジェクトハンドル3050にセットし(ステップ4550)、第2のOSの通信オブジェクトハンドル有効フラグ3060をONにし(ステップ4560)、パラメータで指定されたイベントの名前を通信オブジェクトの名前3020に格納し(ステップ4565)、エントリ有効フラグ3010をONにして(ステップ4570)、そのエントリの記述子を返して終了する(ステップ4580)。

【0063】図20は、第2のOSのプロセス3500が発行したイベント同期通信のウェイト要求システムコ

ールの処理のフローである。このシステムコールは、記述子をパラメータに指定してウェイト要求を行う。まず、当該ウェイト要求において指定された記述子に対応するエントリを仮想クラスタ間通信オブジェクト管理表3000からみつけ、そのエントリにおける第2のOSの通信オブジェクトハンドル3050を用いて、第2のOSのプロセス間通信機能3300のウェイト要求のシステムコールを発行し（ステップ4710）、リターンする（ステップ4720）。

【0064】図21は、第2のOSのプロセス3500が発行したイベント同期通信のポスト要求システムコールの処理のフローである。このシステムコールは、記述子をパラメータに指定してポスト要求を行う。まず、第1のOSへコンテキスト切替えを行い（ステップ4810）、当該ポスト要求において指定された記述子に対応するエントリを仮想クラスタ間通信オブジェクト管理表3000からみつけ、そのエントリにおける第1のOSの通信オブジェクトハンドル3030を用いて、第1のOSのプロセス間通信機能3200のポスト要求のシステムコールを発行する（ステップ4820）。そして、第2のOSへコンテキスト切替えを行い（ステップ4830）、リターンする（ステップ4840）。

【0065】図22は、第2のOSのプロセス3500が発行したイベント同期通信のクローズ要求システムコールの処理のフローである。このシステムコールは、記述子をパラメータに指定してクローズ要求を行う。まず、当該クローズ要求において指定された記述子に対応するエントリを仮想クラスタ間通信オブジェクト管理表3000からみつけ、そのエントリにおける第2のOSの通信オブジェクトハンドル3050を用いて、第2のOSのプロセス間通信機能3300のクローズ要求のシステムコールを発行し（ステップ4910）、第2のOSの通信オブジェクトハンドル有効フラグ3060をOFFにする（ステップ4920）。第1のOSの通信オブジェクトハンドル有効フラグ3040がOFFなら、エントリ有効フラグ3010をOFFにして当該エントリを無効にして（ステップ4940）、リターンする（ステップ4950）。一方、第1のOSの通信オブジェクトハンドル有効フラグ3040がONなら、リターンする（ステップ4950）。

【0066】以上の図15から図22に示した処理により、ウェイトしている第1のOSのプロセス3400を第2のOSのプロセス3500がポストし、ウェイトしている第2のOSのプロセス3500を第1のOSのプロセス3400がポストすることが可能となる。

【0067】

【発明の効果】本発明では、第1の仮想クラスタのオペレーティングシステム（第1のオペレーティングシステム）の起動時に、他の仮想クラスタが利用するハードウェア資源を他の仮想クラスタのために予約して第1のオ

ペレーティングシステムの管理対象から外して第1のオペレーティングシステムからはアクセス不可能にし、これらのオペレーティングシステムから独立したプログラムが外部からの割り込みを扱い、どのオペレーティングシステムで処理するかを決定し、割り込みハンドラを起動することにより、1台の対称型マルチプロセッサシステム（SMP）において複数の仮想クラスタを特別なハードウェアなしに構成することが可能となる。

【0068】これにより、高多重の対称型マルチプロセッサシステム（SMP）において、排他制御オーバーヘッドを小さくする効果がある。また、高多重の対称型マルチプロセッサシステムにおいて、キャッシュミス、およびコヒーレントトランザクションを減らす効果もある。

【0069】また、本発明では、仮想クラスタ間通信オブジェクトの名前と、第1のオペレーティングシステムのプロセス間通信機能におけるプロセス間通信オブジェクトハンドルと、第2のオペレーティングシステムのプロセス間通信機能におけるプロセス間通信オブジェクトハンドルとの対応管理手段を設ける。この対応管理手段により、第1のオペレーティングシステム上のプロセスおよび第2のオペレーティングシステム上のプロセスから発行された仮想クラスタ間通信オブジェクトの名前を指定した通信要求を、第1のオペレーティングシステムあるいは第2のオペレーティングシステムのプロセス間通信機能への要求へと変換できる。これにより、例えば、イベント同期によるプロセス間通信の場合では、ウェイトしている第1のオペレーティングシステム上のプロセスを第2のオペレーティングシステム上のプロセスがポストし、ウェイトしている第2のオペレーティングシステム上のプロセスを第1のオペレーティングシステム上のプロセスがポストすることが可能となる。

【0070】これにより、仮想クラスタ上のプロセス同士が連携して処理を進行することが可能となる効果がある。

【図面の簡単な説明】

【図1】本発明の実施例を実施する計算機システムである対称型マルチプロセッサシステム1000の構成を示す図である。

【図2】ページテーブル1052の構成を示す図である。

【図3】割り込みテーブル1051の構成を示す図である。

【図4】計算機のブート時に実行されるカーネルローダ1131の処理のフローである。

【図5】カーネル構成情報ファイル1600の内容を示す図である。

【図6】主記憶リスト1710、デバイスリスト1720、ロードオブジェクトリスト1740およびパラメータテーブル1700の構成を示す図である。

【図7】割り込みベクタ管理テーブル2000の構造を

示す図である。

【図8】I/Oアドレス管理リスト2100の構造を示す図である。

【図9】プロセッサ管理テーブル2150の構造を示す図である。

【図10】第1のOSの初期化処理のフローである。

【図11】第2のOSのロード処理のフローである。

【図12】割り込み制御装置1080を説明する図である。

【図13】主記憶装置1050における仮想クラスタ間通信関連の内容を示す図である。

【図14】仮想クラスタ間通信オブジェクト管理表3000の図である。

【図15】第1のOSのプロセス3400が発行したイベント同期通信のオープン要求システムコールの処理のフローである。

【図16】第1のOSのプロセス3400が発行したイベント同期通信のウェイト要求システムコールの処理のフローである。

【図17】第1のOSのプロセス3400が発行したイベント同期通信のポスト要求システムコールの処理のフローである。

【図18】第1のOSのプロセス3400が発行したイベント同期通信のクローズ要求システムコールの処理のフローである。

【図19】第2のOSのプロセス3500が発行したイベント同期通信のオープン要求システムコールの処理のフローである。

【図20】第2のOSのプロセス3500が発行したイベント同期通信のウェイト要求システムコールの処理のフローである。

【図21】第2のOSのプロセス3500が発行したイベント同期通信のポスト要求システムコールの処理のフローである。

【図22】第2のOSのプロセス3500が発行したイベント同期通信のクローズ要求システムコールの処理のフローである。

【符号の説明】

1000：対称型マルチプロセッサシステム（SMP）、1010：プロセッサ、1011：プロセッサ、1020：演算装置、1030：割り込みテーブルレジスタ、1040：ページテーブルレジスタ、1045：アドレス変換装置、1050：主記憶装置、1051：割り込みテーブル、1052：ページテーブル、1057：第2のOSの領域、1060：クロック、1070：ブートROM、1080：割り込み制御装置、1090：バス、1100：割り込み信号線、1101：割り込みバス、1110：キーボード、1120：モニタ装置、1130：磁気ディスク装置、1131：カーネルロード、1140：その他の外部装置、1200：仮想ページ番号、1210：有効ビット、1220：物理ページ番号、1300：割り込み番号、1310：割り込みハンドラアドレス、1600：カーネル構成情報ファイル1700：パラメータテーブル、1710：主記憶リスト、1720：デバイスリスト、1730：カーネル構成情報アドレス、1740：ロードオブジェクトリスト、1750：主記憶ブロック、1800：デバイスブロック、1850：ロードオブジェクトブロック、1900：カーネル構成情報、2000：割り込みベクタ管理テーブル、2100：I/Oアドレス管理リスト、2150：プロセッサ管理テーブル、2410：割り込み配送テーブル、2420：割り込み送信装置、2430：選択装置、2440：割り込みマスクレジスタ、3000：仮想クラスタ間通信オブジェクト管理表、3010：仮想クラスタ間通信処理部、3200：第1のOSのプロセス間通信機能、3300：第2のOSのプロセス間通信機能、3400：第1のOSのプロセス、3500：第2のOSのプロセス。

【図2】

図2

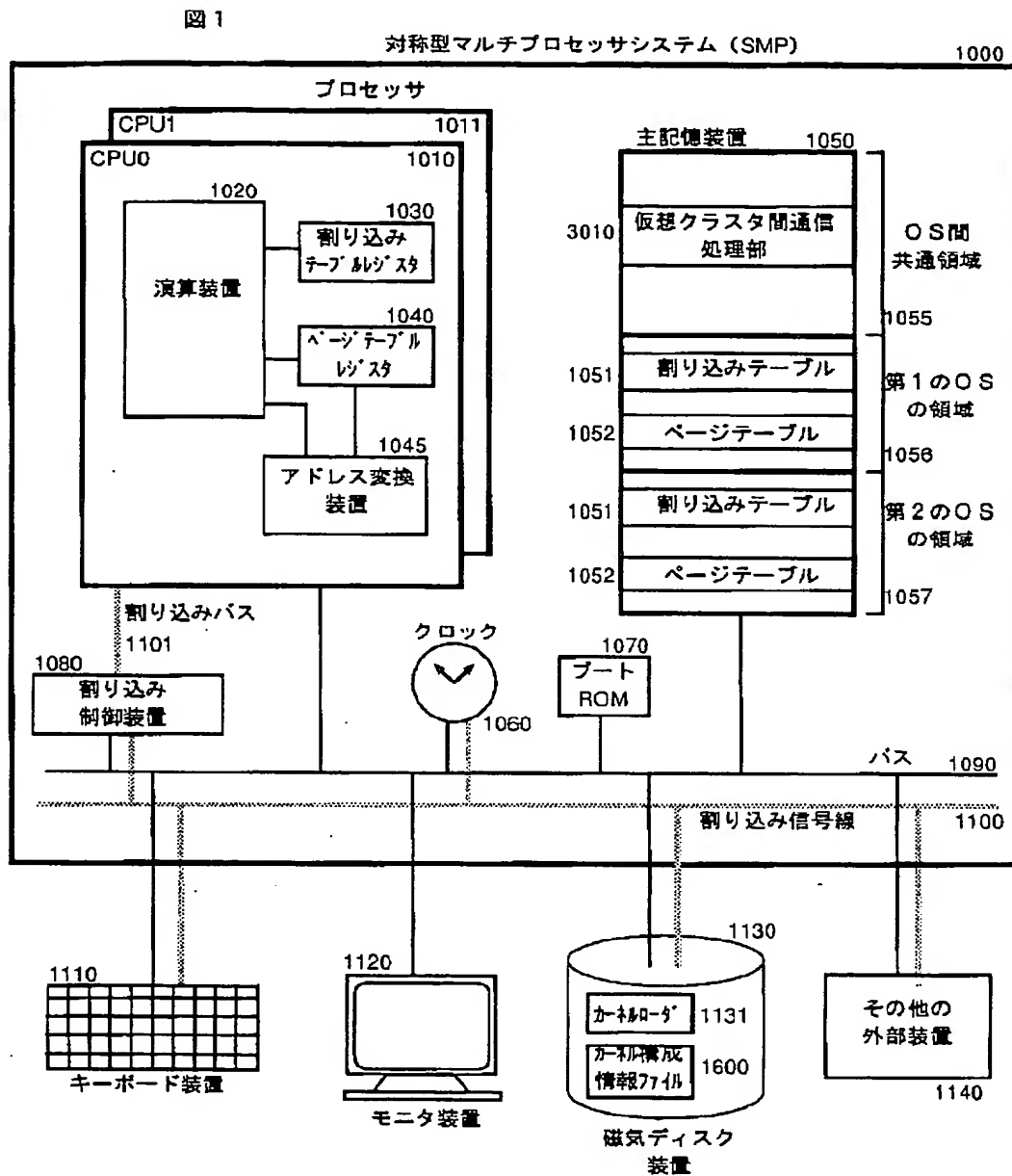
ページテーブル		
仮想ページ番号	有効ビット	物理ページ番号
0	1	0
1	1	3
2	0	28
3	1	15
4	0	41
⋮	⋮	⋮

【図3】

図3

割り込みテーブル	
割り込み番号	割り込みハンドラアドレス
0	200
1	220
2	240
3	260
4	280
⋮	⋮

【図1】



【図5】

図5 カーネル構成情報ファイル 1600

名前	内容
1610 オブジェクトファイル	kernel, driver 1630
1620 第2のOS	第2のOSが使用するデータ 1640

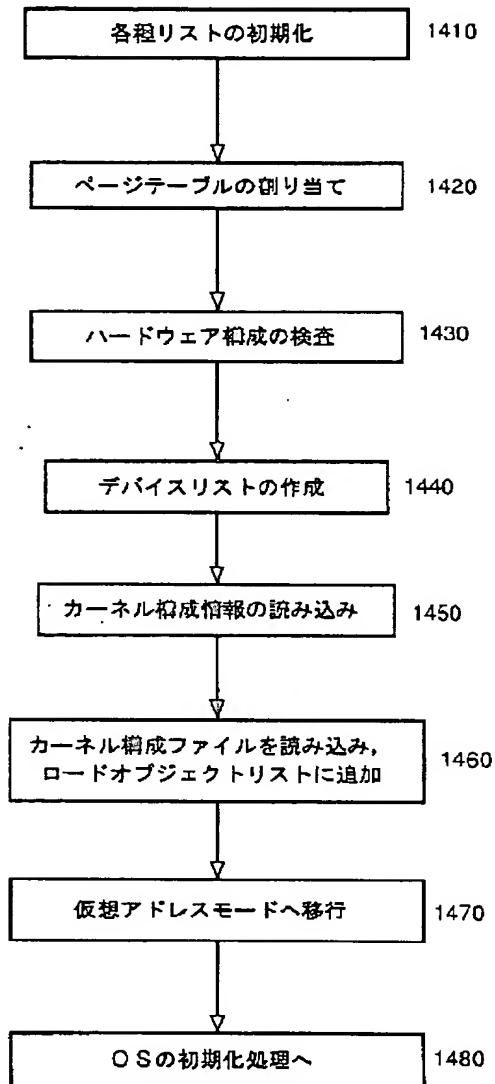
【図9】

図9 プロセッサ管理テーブル 2150

プロセッサ	使用中
CPU0	
CPU1	

【図 4】

図 4



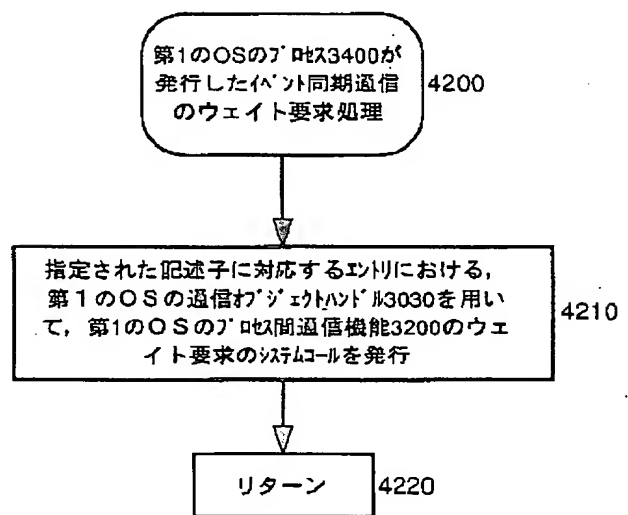
【図 7】

図 7

割り込み番号	使用中
0	
1	
2	
3	
4	
⋮	

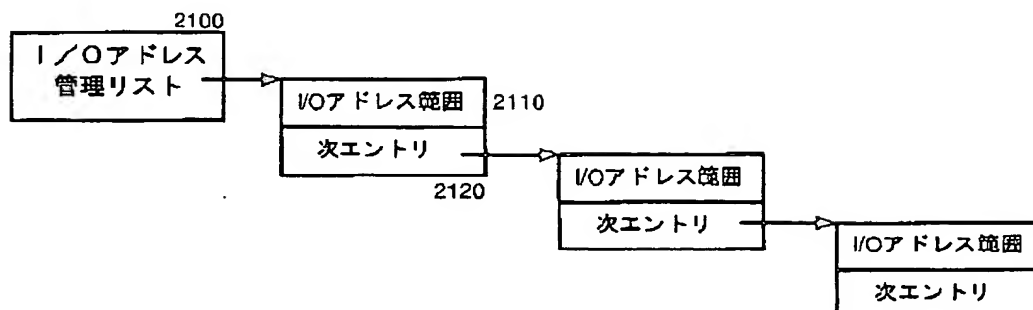
【図 16】

図 16

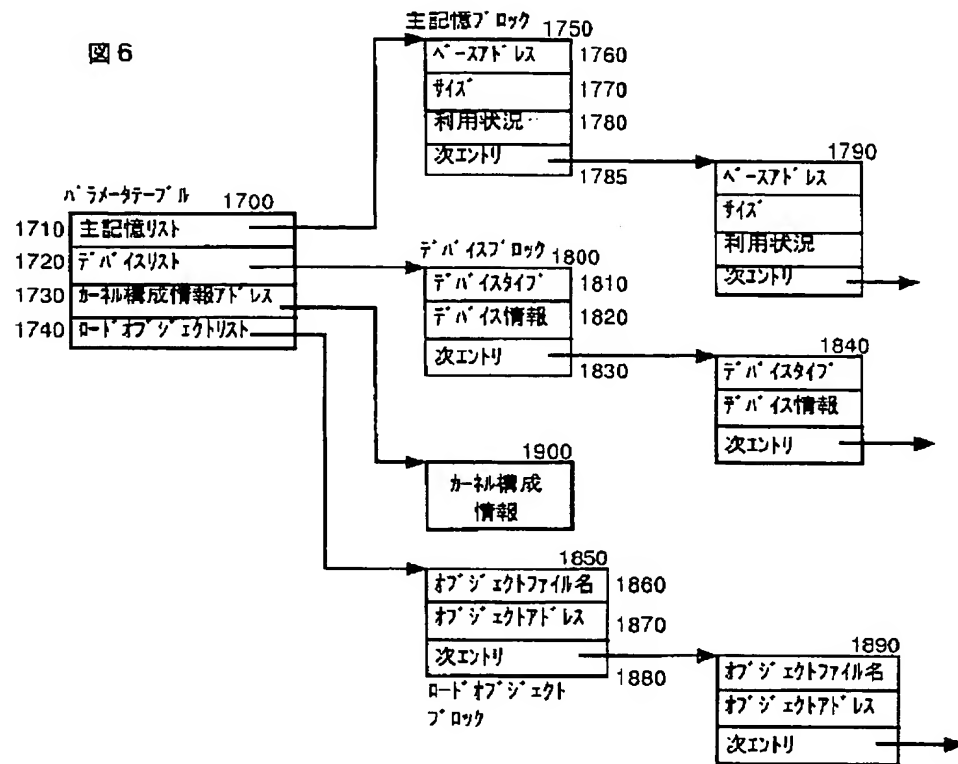


【図 8】

図 8



【図6】



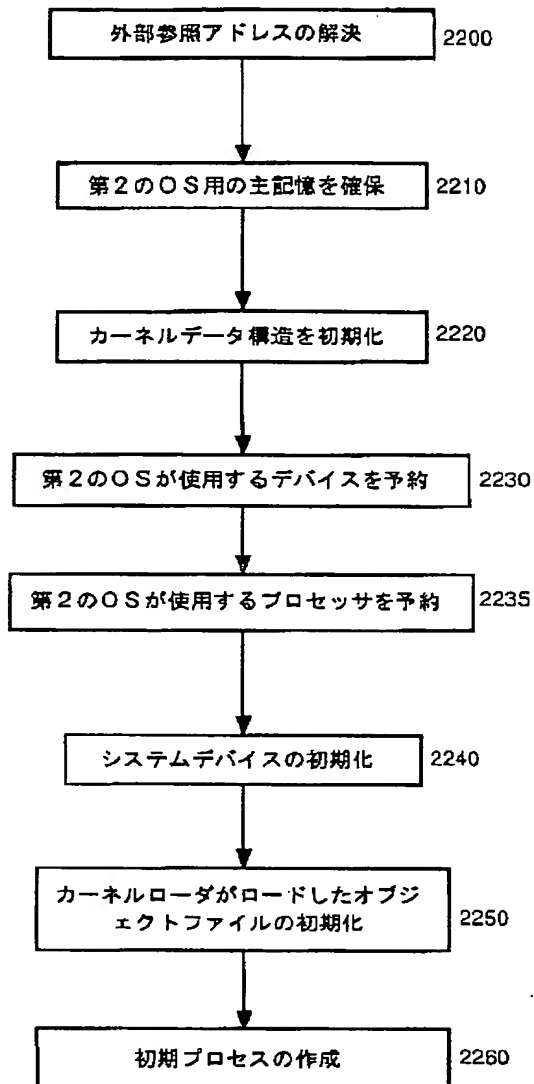
【図14】

図14

仮想クラス間通信オブジェクト管理表						3000
エントリ有効フラグ	通信オブジェクトの名前	第1のOSの通信オブジェクトハンドル	第1のOSの通信オブジェクトハンドル有効フラグ	第2のOSの通信オブジェクトハンドル	第2のOSの通信オブジェクトハンドル有効フラグ	
3010	3020	3030	3040	3050	3060	

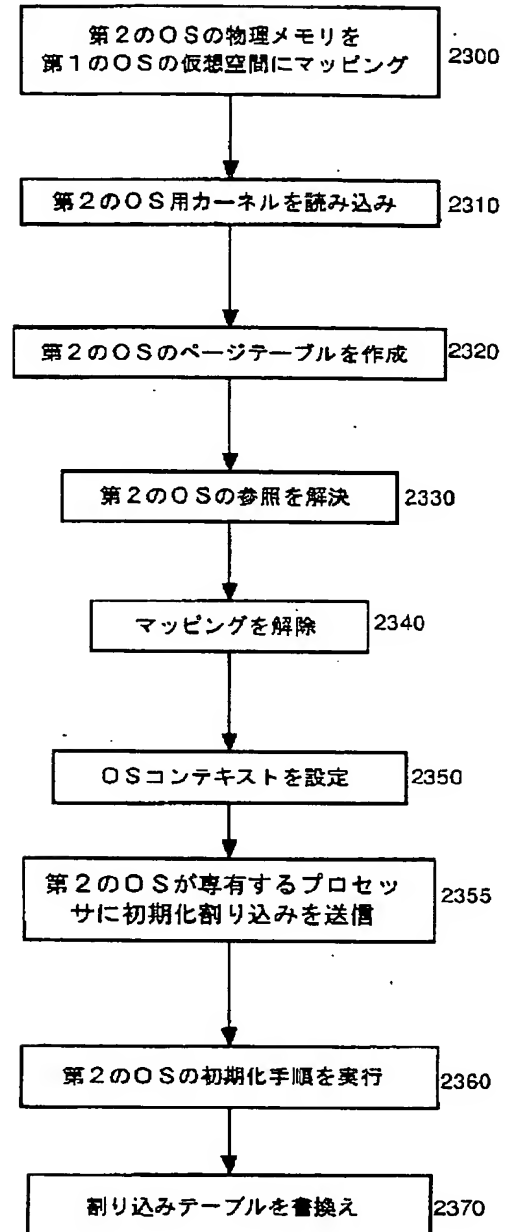
【図10】

図10



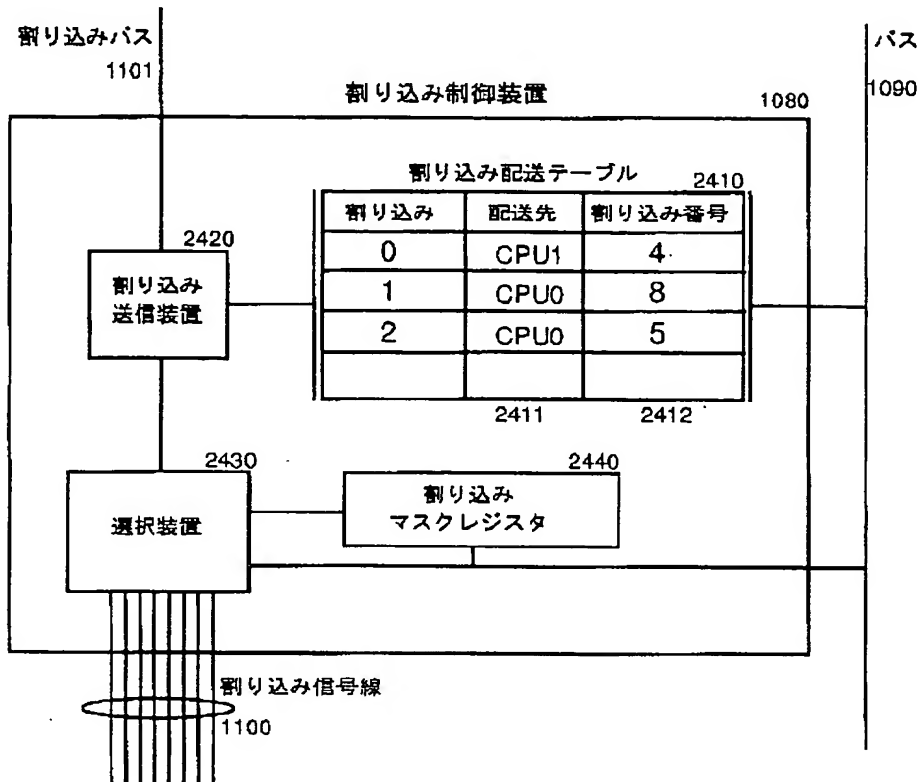
【図11】

図11



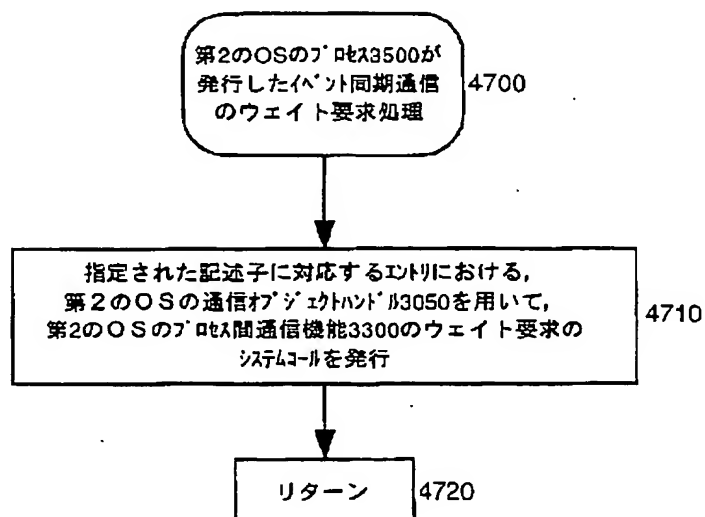
【図12】

図12



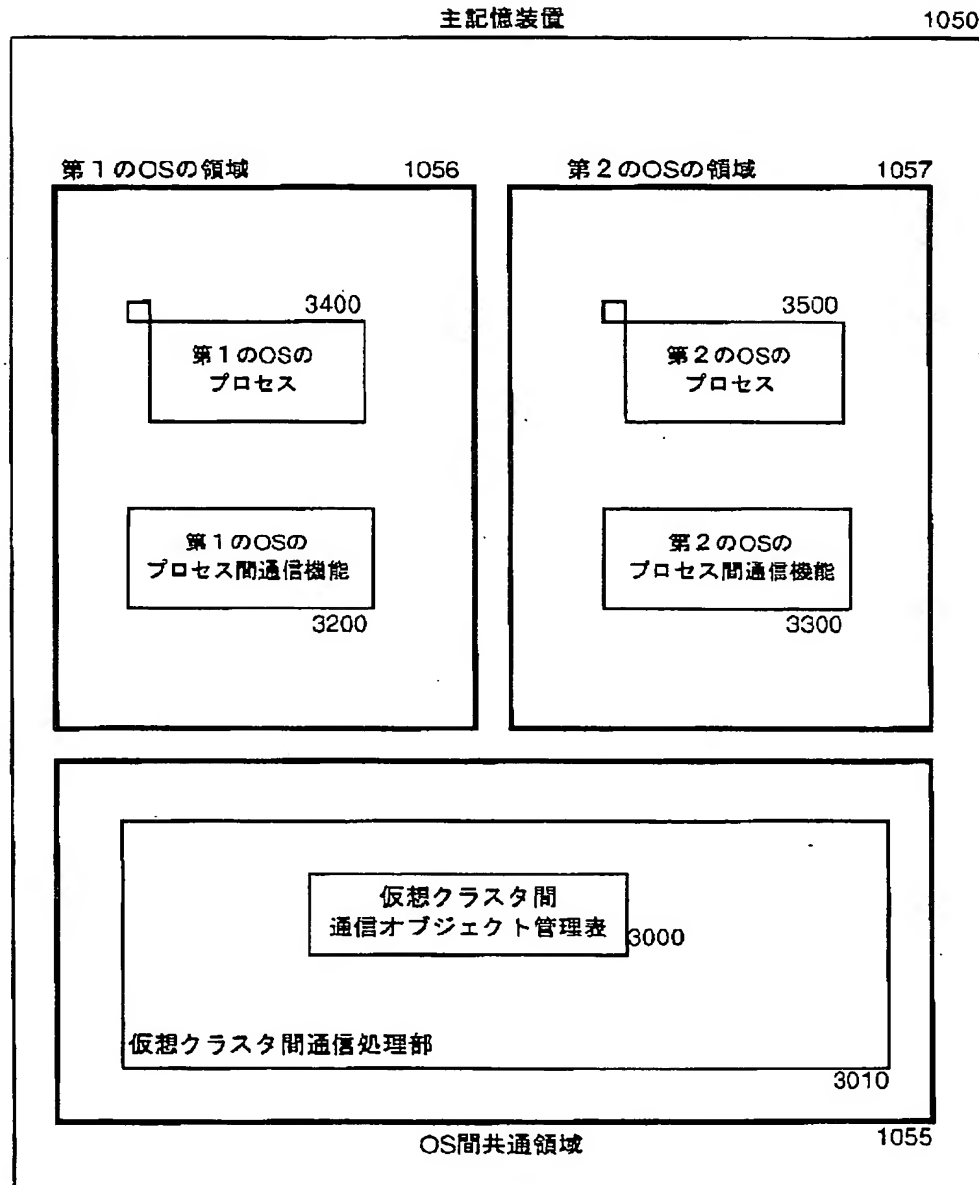
【図20】

図20



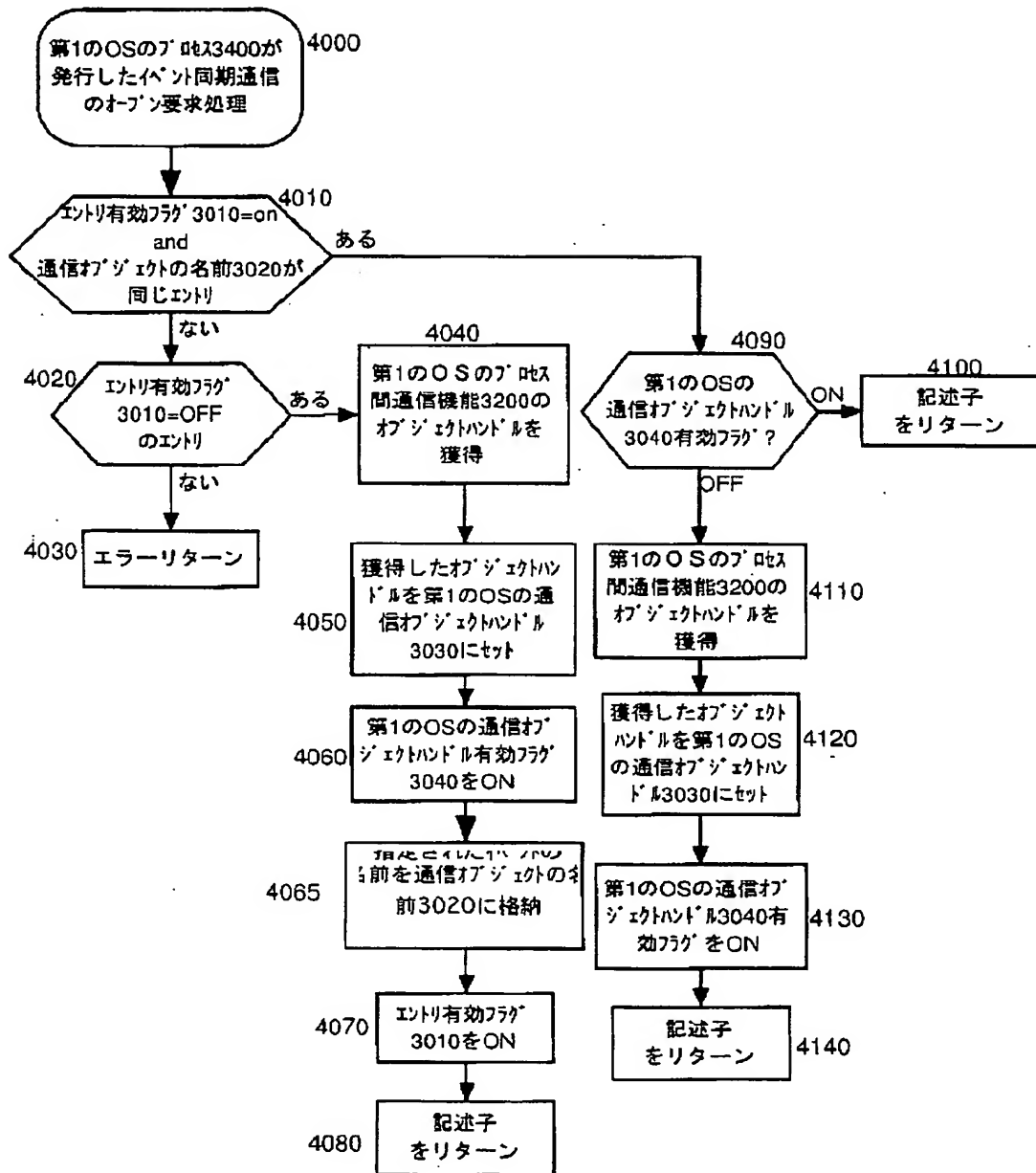
【図13】

図13



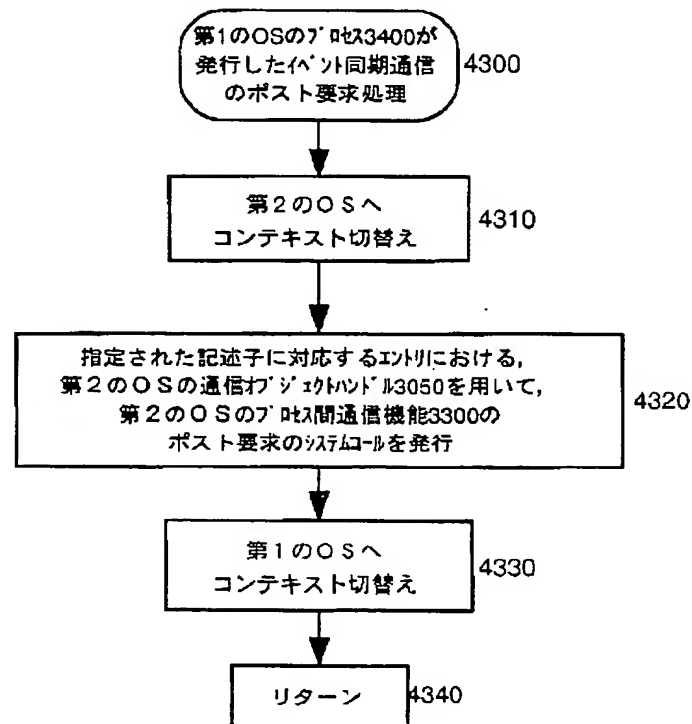
【図 1 5】

図 1 5



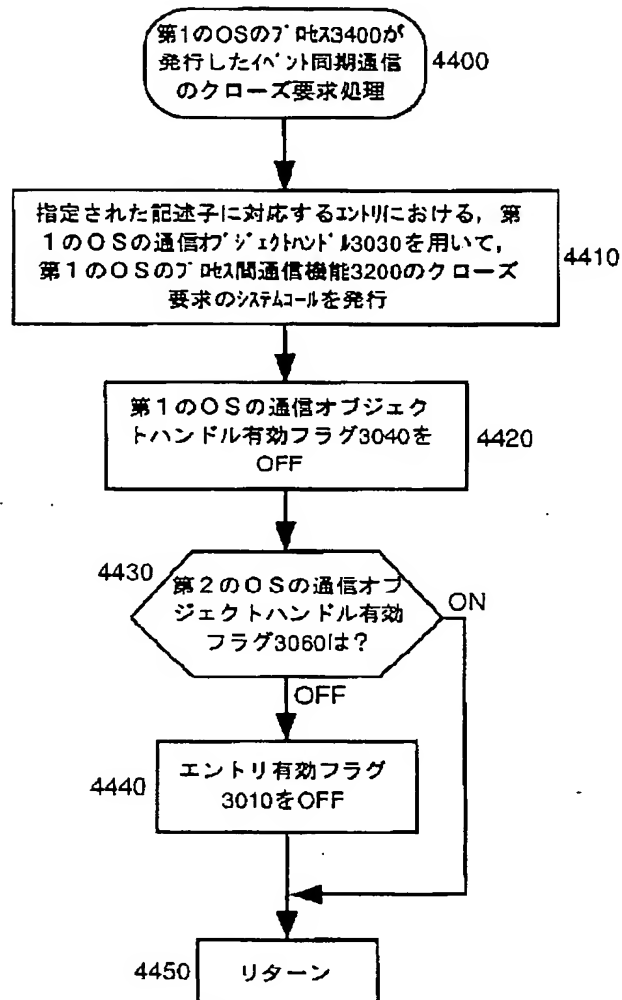
【図17】

図17



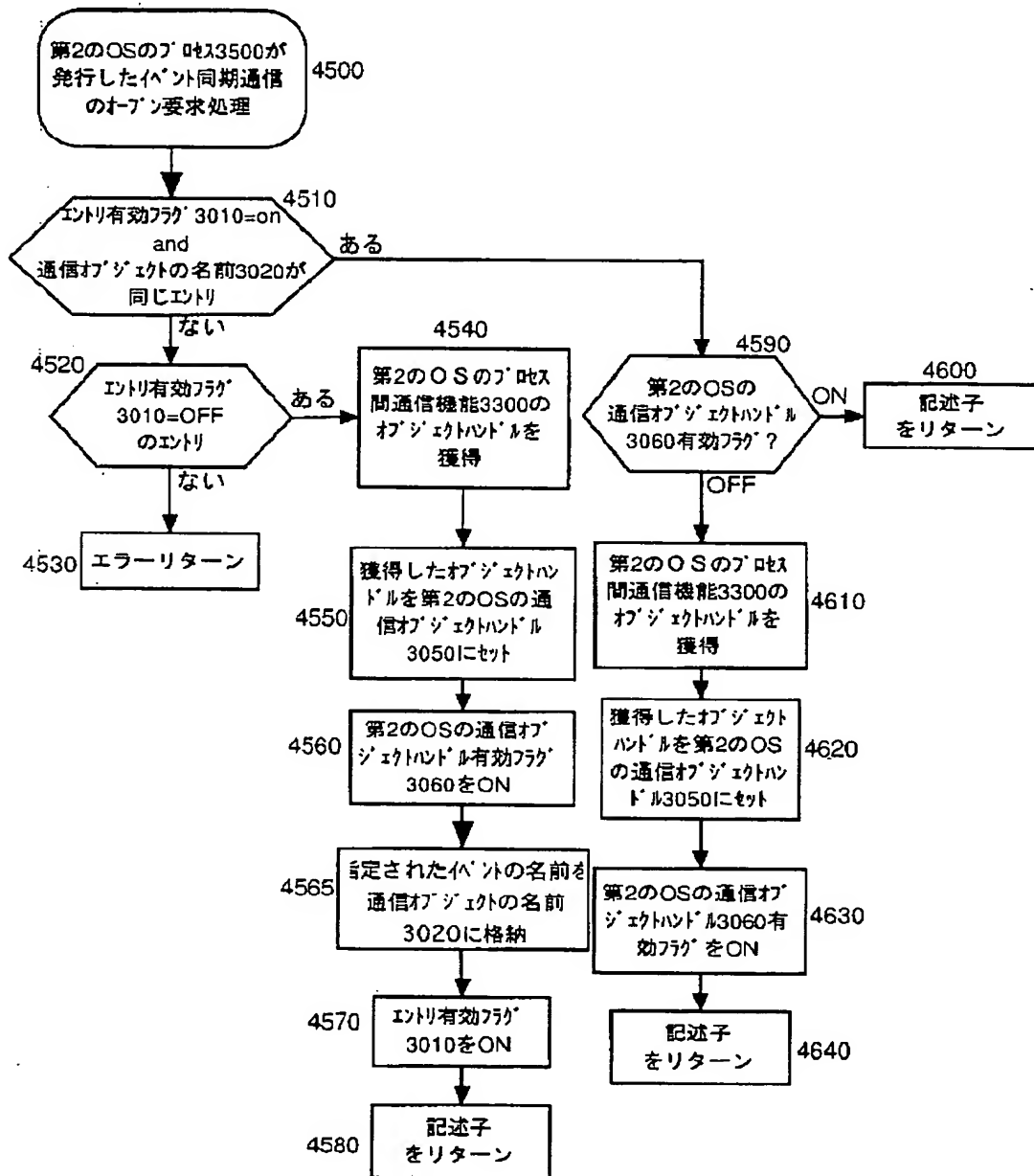
【図18】

図18



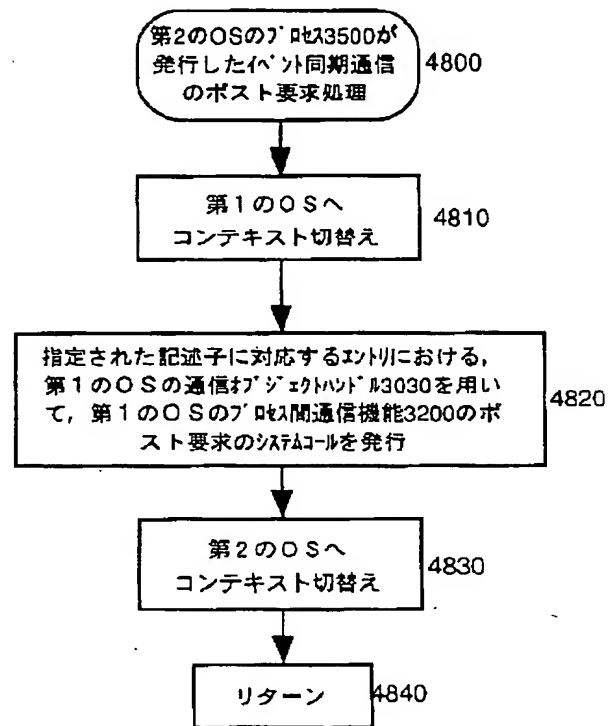
【図 19】

図 19



【図 21】

図 21



【図 22】

図 22

